Fuzzy function approximation with guaranteed error-bounds estimate

V. Vigneron, H. Maaref, C. Barret and R. Kallel

Abstract— **Usually, fuzzy systems approximate functions by covering their graphs with fuzzy patches in the input-output state space. Each fuzzy rule defines a fuzzy patch. The approximation increases in accuracy as the fuzzy patches increase in number and decrease in size. In this paper, we propose an other approach for fuzzy approximation in which fhe estimation of the *fuzzy parameters* from experimental data is viewed as one of set invertion, which is solved in an approximate but guaranteed way with the tools on interval analysis. Is is, for instance, possible to characterize the set of all parameters vectors that are consistent with the data in the sense that the errors between the data and corresponding model outputs fall within known prior bounds. Any prior knowldge that can be expressed as a series of inequalities to be satisfied by the parameters can be taken into account. The purpose of this paper is to briefly present some results recently obtained in the field of fuzzy approximation.**

Keywords— **Fuzzy sets, interval computation, nonlinear estimation, set inversion.**

In the following real numbers and fuzzy numbers are denoted by lowercase letters and uppercase letters, respectively.

I. Fuzzy function approximation

A fuzzy rule-based model is a set of fuzzy If-Then rules that maps inputs to outputs. So it defines a function $f : X \to Y$. A fuzzy system contains a set of rules of the form

If X is A Then Y is B.

Fuzzy rules define fuzzy patches or subsets of the state space $X \times Y$. Less certain rules are large patches. More precise are small patches. Figure 1 shows how fuzzy patches in the input-output product space $X \times Y$ cover the real function f. The association "If $X = A_2$ Then $Y = B_2$" defines a patch in $X \times Y$. The patch is fuzzy since A_2 and B_2 are fuzzy sets. A three dimensional plot would show the fuzzy patch as a barn-like structure rising up from its eliptic base. In the figure 1, the collection of all fuzzy cartesian products defines 9 overlapping fuzzy patches.

Usually, experts state the fuzzy rule or a neural or statistical procedure generates them from sample data, when it is too hard or impossible for human beings to give the desired fuzzy rules or membership functions, due to ambiguity, uncertainty or complexity of the identifying system. In this case, it is natural and necessary to generate or tune fuzzy rules by some learning technique: so-called *neuro-fuzzy learning algorithms* have been proposed by Ichihashi [4], Nomura *et al.* [12], Wang and Mendel [18] independently.

V. Vigneron is with the CEMIF-LSC FRE 2494, 40 rue du Pelvoux, 91020 Evry Cedex, France. E-mail: vvigne@cemif.univ-evry.fr and with the MATISSE-SAMOS UMR 8595, 90, rue de Tolbiac, 75634 Paris cedex 13, France. E-mail: vigneron@univ-paris1.fr

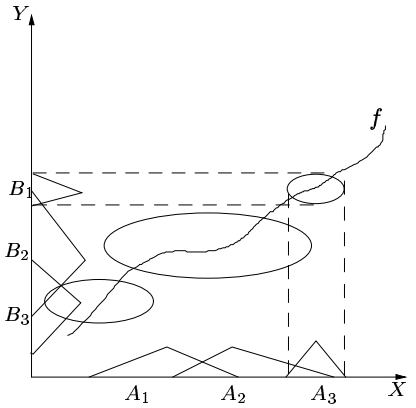


Fig. 1. The fuzzy rule patch "If $X = A_2$ Then $Y = B_2$" is the fuzzy cartesian product $A_2 \times B_2$ in the input-output product space $X \times Y$.

Most methods for extracting fuzzy rules from numerical data assume the divisions of input variables into fixed regions (see Abe *et al.* [1] and Bauman *et al.* [2]). Nonlinear systems were described by Sugeno [14] as a collection of linear subsystems with variable parameters depending on the measure of belonging of the state vector to different fuzzy regions, generally not disjunct, defined by If-Then logical rules. His major idea was to use a number of models which are effective around the respective operating points. The major disadvantages of such multi-model approach lie in partitioning data, determining membership functions and selecting explanatory variables. These tasks are so strongly related to each other that they make the modeling quite difficult. See also TS fuzzy reasonning proposed by Takagi and Sugeno [14], [15], [19]. Dickerson and Kosko [6] approximate any function with ellipsoidal rules: statistical clustering algorithm convert the sample data $(x_i, y_i), i = 1, \ldots, n$, n being the number of input-output pairs, into cluster estimates. Clustering algorithms search for the implicit fuzzy rules that the "physical process" used to generate the data: in other words, the fuzzy system learns or adapts locally to fit a function with training data.

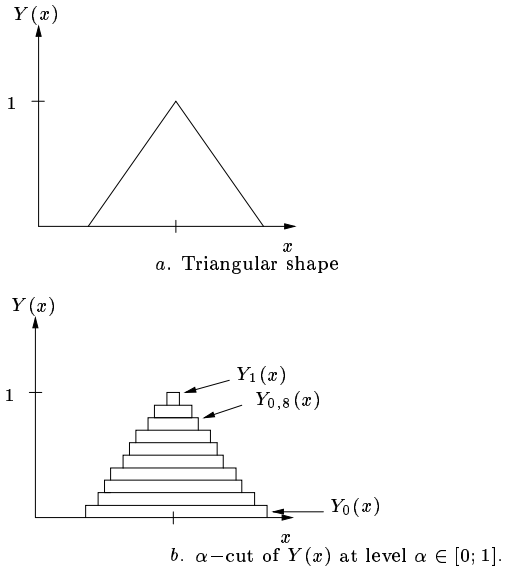
The purpose of this paper is to advocate a promising alternative approach which, to our best knowledge, has never been used in the context of fuzzy system. The approach uses set-membership estimation (see e.g. [10] and references therein). The learning task is formulated here as a constrained optimization problem where a family of functions $\mathcal{F}$ has to be found to relate some predefined fuzzy sets A and B using interval computation. This approach encompasses all the acceptable values of the parameter vector in a set that is characterized. It leads to a simple algo-

rithm that tunes the function parameters depending on the size of the patches $A \times B$. In the precise limit the rule patch $A \times B$ is a point if A and B are binary spikes (in this case, $f \in \mathcal{F}$ could be approximated using classical estimation techniques). The advantages of this approach are twofold: no statistical assumption on the modeling error is required, and any bounded error can be treated independently from its origin (modeling and/or measurement error).

Our general formulation of the fuzzy approximation problem is presented in section II. Section III recalls the basic notions of interval analysis to be used. The practical significance of the computational results is pointed out through section III-C.0.a with the help of some numerical examples.

II. Formulation of the fuzzy approximation problem

A fuzzy rule approximates a function by defining a fuzzy patch in the state space. A fuzzy rules relates fuzzy sets. The simplest fuzzy rule associates output fuzzy set B with an input fuzzy set A and defines a fuzzy transformation f that maps similar input fuzzy sets A' to similar output fuzzy sets B'. The fuzzy rule f equals the fuzzy cartesian product patch $A \times B$. The rules and their sets can have any shape. In practice, fuzzy sets have simple shapes like trapezoids, bell curves or triangles (see Fig.2.a for an example).



a. Triangular shape

b. $\alpha-$cut of $Y(x)$ at level $\alpha \in [0;1]$.

Fig. 2. Representation of fuzzy set Y.

A rule changes when its fuzzy sets change. A fuzzy set expresses the degree to which an element belongs to a set: if $\mathcal{Y}$ is a collection of elements generally denoted by y, a fuzzy set Y in $\mathcal{Y}$ is defined as a set of ordered pairs $\{(y, \mu_Y(y)) \mid y \in \mathcal{Y}\}$, where $\mu_Y(y)$ is called the *membership function* for the fuzzy set Y. If the value of the membership function is restricted to either 0 or 1, then Y is "reduced" to an ordinary set. Given a fuzzy set Y defined on $\mathcal{Y}$ and any number α in the unit interval $[0,1]$, the so-called α-cut of Y, denoted by Y^α is a crisp set that consists of $\{y \mid \mu_Y(y) \geq \alpha\}$[1]. This can formally be written as

$$\mu_Y^\alpha = \alpha \mu_Y(y), y \in \mathcal{Y}. \tag{1}$$

When Y is defined on the set of real numbers $\mathbb{R}$, μ_Y^α are called the *characteristic functions*, and are defined for each $\alpha \in [0,1]$ by

$$\mu_Y^\alpha(y) = \begin{cases} \alpha & \forall y \in Y^\alpha \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

with $Y = \cup_{\alpha \in [0,1]} Y^\alpha$, such as

$$\left(\cup_{\alpha \in [0,1]} Y^\alpha\right)(y) = \sup_{\alpha \in [0,1]} \mu_Y^\alpha(y), \forall y \in \mathcal{Y}. \tag{3}$$

Each fuzzy set is a collection of nested crisp sets that are conceived as a whole. In Figure 2.b, a collection of crisp sets which defines the characteristic function μ_Y^α approximates the membership fonction μ_Y at level $\alpha \in [0,1]$.

The approximation $f : X \to Y$ increases as the fuzzy patches decrease in size, *i.e.* as $\alpha \to 1$ or when we add more small patches, the counterpart being that the complexity costs increase. By definition, a fuzzy system that learns is *any model* such that parameters, inputs and targets are given as fuzzy numbers. In computer simulations, fuzzy arithmetic is approximately performed by interval arithmetic on α-level sets of each fuzzy number [5]. In this paper, we used 10 level sets (*i.e.* $\alpha = 0, 0.1, \ldots, 1$). As shown in Figure 1, the antecedent fuzzy sets $(A_1, A_2, \ldots, A_p)$ of each fuzzy If-Then rule are used as an input vector, and the consequent fuzzy sets $(B_1, B_2, \ldots, B_s)$ are used as the corresponding target vector in the learning. We assume in the following, for sake of simplicity, that the linguistic values are specified by triangular membership functions. It should be noted that a single input (or output) unit handles a fuzzy input (or target) in our model. This is different from other models where for example, multiple input (or output) were employed for handling a single fuzzy input (or target) [5]. Since the task of our fuzzified model is to approximately realize the fuzzy If-Then rules, the consequent fuzzy sets $(B_1^\alpha, B_2^\alpha, \ldots, B_s^\alpha)$ of each rule are used as a fuzzy target vector when the antecedent fuzzy sets $(A_1^\alpha, A_2^\alpha, \ldots, A_p^\alpha)$ are used as a fuzzy input vector. The learning of our fuzzified model consists usually to minimize the difference between the fuzzy target vector $(B_1^\alpha, B_2^\alpha, \ldots, B_s^\alpha)$ and the actual fuzy output vector $f(A_1^\alpha, A_2^\alpha, \ldots, A_p^\alpha)$. The most classical approach for parameter estimation is to look for the value of the parameter vector $\vec{\theta}$ that is best in the sense of a given fuzzy criterion J. One may for instance look for the values of $\vec{\theta}$ that minimizes

$$J(\alpha, \vec{\theta}) = \frac{1}{2} \left\{ \sum_{i=1}^{s} (B_i^{\alpha+} - f(A_i^{\alpha+}, \vec{\theta}))^2 + \sum_{i=1}^{s} (B_i^{\alpha-} - f(A_i^{\alpha-}, \vec{\theta}))^2 \right\}, \tag{4}$$

[1]Similarily, the strong $\alpha-$cut, denoted by $Y^{\alpha+}$, is defined as $\{y \mid \mu_Y(y) > \alpha\}$.

where the upperscripts $^+$ and $^-$ denote the upper and lower limits of the α-level sets respectively. This cost function was first proposed by Krishnamraju *et al.* [7]. A gradient descent learning algorithm is proposed in Ishibuchi *et al.* [5].

Usually the gradient descent algorithm is used for adjusting each parameter value. Such a local optimization of a scalar criterion has several drawbacks:

1. the choice of a initial value for the parameters relies largely on guesswork
2. No guarantee of convergence to the global optimum of the criterion can be provided
3. if there are several values of the estimated parameters that correspond to the same value of the criterion, a situation that may for instance result from the fact that the parameters are not globally identifiable, the algorithm picks one of them withour indicating that there are others
4. in fuzzy problems, one is not actually interested in the optimal value of the parameters in the sense of a criterion but would rather like to characterize the set of all values that are acceptable in a sense to be specified

A possible way out of problems 1 to 3 is to use deterministic global optimization methods such as those described in Belforte *et al.* [3], but this still leaves difficulty 4. This is why we shall follow a different route, and look for the set of all models that are acceptable instead of looking for the optimal model.

A. Core of the modeling

In our learning algorithm, each fuzzy IF-THEN rule is viewed as a fuzzy input-ouput pair. Since the $\alpha-$levels sets of fuzzy members are intervals, each fuzzy input-output pair can be viewed as a collection of interval input-output pairs to be used in the learning of the fuzzified model as inputs and targets.

pattern index	input $[x_i]$	output $[y_i]$
1	[2;6]	[0;2]
2	[1;5]	[1;3]
3	[1;5]	[2;4]
4	[0;4]	[5;7]
5	[-1;3]	[6;8]

TABLE I

EXAMPLES OF INTERVAL INPUT-OUTPUT PAIRS USED ($[x_i], [y_i]$) IN THE LEARNING OF THE FUZZIFIED MODEL.

For example, let us assume that we have a fuzzy IF-THEN rules problem which can be handled as a fuzzy input-output pair $([x_i], [y_i]), i = 1, \ldots, 5$. In Table I, we show the interval input-output pairs generated from some fuzzy input-output pair. The interval input-output pair are also illustrated on Figure 3. Such interval input-output pairs are used in the learning of the fuzzified model as inputs and targets. The procedure employed consists of estimating the fuzzy parameters associated.

In our learning algorithm, first the value of α and the pattern index are set as $\alpha = 0$ and $p = 1$. So the interval $[2; 6]$ is presented as an input. Next the corresponding interval output from the output unit is calculated by interval arithmetic (see following Section III). As shown in Table I, the corresponding target is the interval $[0; 2]$. Then the fuzzy parameters are modified based on the difference e_i between the actual interval output and the target interval. These procedures are iterated for $\alpha = 0, 0.1, \ldots, 1.0$ and $i = 1, \ldots, p$ (here $p = 5$, *i.e.* 5 fuzzy IF-THEN rules).

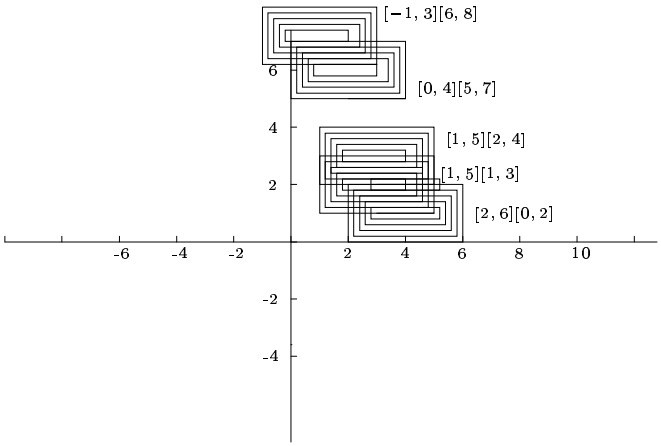


Fig. 3. α-cut representation of some fuzzy input-output pairs.

B. Problem formulation

Let f be some appropriate function. The fuzzy system is assumed to be described by

$$y_i = f(x_i; \vec{\theta}), \quad i = 1, \ldots, p \tag{5}$$

where the observation variable $y_i \in [y_i]$ is related to $x_i \in [x_i]$, a vector of the experiment designed from any linguistic values. It is assumed that f is continuous and differentiable. It is also assumed that domains $[x_i]$ and $[y_i]$ are available. T hese domains can be arbitrarily large, e.g. $[x_i] =]-\infty; \infty[$ if no information is available on a measurement x_i.

In the context of bounded error estimation (e.g. [9] and the references therein), the feasible observation error e_i will be expressed now in terms of a set, *i.e.* $e_i \in \mathbb{E}_i$ where

$$\mathbb{E}_i = \{e_i \in \mathbb{R} \mid e_i^- \leq e_i \leq e_i^+\}. \tag{6}$$

and $i = 1, \ldots, p$ denotes the index pattern. In this equation, e_i^+ and e_i^- are the lower and upper bounds on the observation error, assumed to be known. Estimating the vector $\vec{\theta}$ ($\vec{\theta} \in \Theta \subset \mathbb{R}^m$) of m unknown parameters in this context amounts to looking for the set $\mathbb{S}$ of all admissible values of $\vec{\theta}$ that are consistent with equations (5) and (6). Θ is the prior feasible set for the parameters. $\mathbb{S}$ is thus the intersection of Θ and the set of the solutions for $\vec{\theta}$ of the set of inequalities:

$$y_i - e_i^- \leq f(x_i, \vec{\theta}) \leq y_i - e_i^+, \tag{7}$$

This bounded-error approach leads to set estimates, contrary to usual applications of the maximum-likelihood approach which yields *point estimates*. $\mathbb{S}$ is called the posterior feasible parameter set.

The algorithms used for characterising $\mathbb{S}$ depends on whether the model output f is linear in the parameters. In the first case, it is possible to use an exact description [17]. When the model is nonlinear in the parameters, the problem is much more difficult since $\mathbb{S}$ may be non-convex and even nonconnected. But it is still possible to get a precise description of $\mathbb{S}$ using interval analysis and set-inversion.

III. Tools

A. Basics for set inversion

Let us first very briefly recall few notions of interval computation (see e.g. [8]) needed to present it.

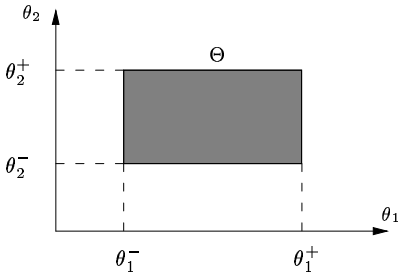


Fig. 4. Example of a 2-dimensional patch with lower and upper bound values.

A *vector interval* (or *box*) in a m-dimensional space is the cartesian product of m scalar intervals (see Figure 4) and is noted

$$[\theta] = [\theta_1^-, \theta_1^+] \times \ldots \times [\theta_m^-, \theta_m^+],$$

where θ_1^- and θ_1^+ denote respectively the lower and upper bound on the interval θ_1, etc. Its ith interval component $[\theta_i]$ is the projection of $[\vec{\theta}]$ onto the ith axis. The width $w([\vec{\theta}])$ of the box $[\vec{\theta}]$ is the length of its largest side(s).

The task of finding all $\vec{\theta} \in \Theta$ satisfying (7) may be expressed as that of characterizing the set

$$\mathbb{S} = \{\vec{\theta} \in \Theta : f(x, \vec{\theta}) \in y - \mathbb{E}\}. \tag{8}$$

$\mathbb{S}$ may be equivalently expressed as

$$\mathbb{S} = f_{\Theta}^{-1}(y - \mathbb{E}) = f_{\Theta}^{-1}(\mathbb{Y}), \tag{9}$$

where f_{Θ}^{-1} is the reciprocal function (in a set theoretic sense) of f defined over Θ and $\mathbb{Y}$ is the prior feasible set for the model outputs. For a given function f, interval analysis provides an *inclusion function* $\mathcal{F}$ that return boxes that are guaranteed to contain the image by f of any given box $[\vec{\theta}]$ included in the domain of f. $\mathcal{F}([x])$ is a box such that

$$f([x], [\vec{\theta}]) \subset \mathcal{F}([x], [\vec{\theta}]),$$

and $w([\vec{\theta}]) \to 0 \Rightarrow w(\mathcal{F}([x])) \to 0$. The last equation is only needed to ensure convergence. It means that the smaller the box $[\vec{\theta}]$ is, the better the approximation provided by the inclusion function is going to be. The analysis of the parameter space will be performed by building sets of non-overlapping boxes with nonzero width (or

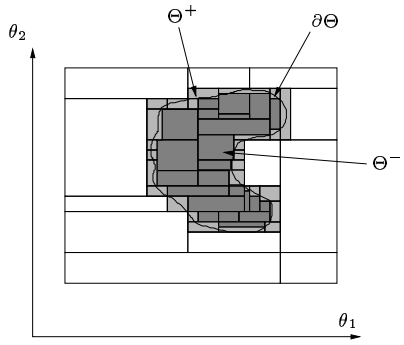


Fig. 5. Bracket of the portion of X contained in $[\breve{\vec{\theta}}]$.

subpavings). Exploration starts from an initial box of interest $[\vec{\theta}](0)$ which is split by the algorithm into smaller boxes whenever needed until the width of the box becomes smaller than some tolerance parameter ϵ_r to be specified by the user. Interval computation provides us 2 basic tests for deciding whether a given box $[\vec{\theta}]$ is included in $\vec{\theta}$:

$$\mathcal{F}([x], [\vec{\theta}]) \subset [y] \Rightarrow [\vec{\theta}] \in \Theta (\text{ i.e. } [\vec{\theta}] \text{ is feasible})$$
$$\mathcal{F}([x], [\vec{\theta}]) \cap [y] = \varnothing \Rightarrow [\vec{\theta}] \cap \Theta = \varnothing (\text{ i.e. } [\vec{\theta}] \text{ is } \textit{unfeasible})$$

In all other cases, $[\vec{\theta}]$ is *indeterminate*. If none of these 2 conditions is satisfyed, our algorithm computes two subpavings iteratively, namely Θ^- containing all boxes that were proved feasible, and Θ^+ consisting of all indeterminate boxes (see Figure 5). $\forall\Theta$, it is always possible to find 2 finite subpavings Θ^- and Θ^+ such that $\Theta^- \subset \Theta \subset \Theta^+$. From these subpavings, it is easy to bracket the portion of Θ contained in $[\vec{\theta}](0)$ as:

$$\Theta^- \subset [\vec{\theta}](0) \cap \Theta \subset \Theta_{out} = \Theta^- \cup \Theta^+. \tag{10}$$

Since Θ_{out} is a finite union of boxes guaranteed to contain the portion of Θ of interest, it is very convenient for implementing set-theoretic manipulations [9]. A stack will be used to store the boxes still under considerations.

B. Θ-set characterization

The algorithm requires a possibly very large search box $[\vec{\theta}](0)$ to which Θ^+ is guaranteed to belong. 4 cases may be encountered:

1. if $\mathcal{F}([x], [\vec{\theta}](k)$, has no empty intersection with $[y]$, but is not entirely in $[y]$, then $[\vec{\theta}]$ may contain a part of the solution. $[\vec{\theta}]$ is said to be *undetermined*. If it has a width greater than a prespecified precision parameter ϵ_r, then it should be bisected and the test should be recursively applied to these newly generated boxes.
2. if $\mathcal{F}([x], [\vec{\theta}](k))$ has an empty intersection with $[y]$, then $[\vec{\theta}]$ does not belong to Θ
3. if $\mathcal{F}([x], [\vec{\theta}](k))$ is entirely in $[y]$, then $\vec{\theta}$ belongs to the solution subpaving Θ and is stored in Θ^- and $\vec{\theta}^+$.
4. if the box considered is undetermined, but its width is lower than ϵ_r, then it is deemed small enough to be stored in the outer apporximation Θ^+ of Θ.

Upon completion of this algorithm, no indeterminate box will have a width larger than ϵ_r. Under a few realistic technical conditions Θ^+ and Θ_{out} will tend to Θ (respectively from within and from without) when $\epsilon_r \to 0$ [13]. At the end, the set Θ^- of all boxes that have been proved to be feasible can be plotted in the parameter space (see Figure 5).

1.	if $\mathcal{F}([x],[\vec{\theta}](k)) \cap \mathbb{Y} = \varnothing$, return;
2.	if $\mathcal{F}([x],[\vec{\theta}](k)) \subset Y$, then $\{\Theta^- = \Theta^- \cup [\vec{\theta}];\ \Theta^+ = \Theta^+ \cup [\vec{\theta}]\}$, return;
3.	if $w([\vec{\theta}](k)) \leq \epsilon_r$, then append $\{\Theta^+ = \Theta^+ \cup [\vec{\theta}];\ \Theta^+ = \Theta^+ \cup [\vec{\theta}]\}$, return;
4.	if the stack is not empty, then unstack into $[\vec{\theta}](k+1)$, increment k by one and go to the step 1, else stop.

TABLE II

ALGORITHM BASED ON A INCLUSION FUNCTION.

The algorithm summarized by Table II is a recursive algorithm, where the subpavings Θ^- and Θ^+ have been initialized as empty, and by setting $k = 0$. Calculus performed by Jaulin and Walter [9] show that the algorithm terminates after less than

$$\left(\frac{w([\vec{\theta}](0))}{\epsilon_r} + 1\right)^n \tag{11}$$

the bisections and the computing time increasing exponentially with the dimension of $\vec{\theta}$.

The subpaving $\partial\Theta \triangleq \Theta^+ \backslash \Theta^-$ consisting of all boxes of Θ^+ that are not in Θ^- is called the uncertainty layer. It is a regular subpaving whose boxes have a width smaller than ϵ_r.

Provided that Θ is full, this means that the pair $[\Theta^-; \Theta^+]$ defines a neighbourhood of Θ with a diameter that can be chosen arbitrary small, as illustrated on Figure 6. In this Figure, subpavings have been generated to bracket the posterior feasible set $\vec{\theta}$ for the parameters between inner and outer approximations. Such subpavings are simpler to store or manipulate than membership functions μ_X^α or μ_Y^α, but form an expansive representation of the fuzzy approximation problem in terms of memory space. They are thus well adapted to low dimensional problems [13], [17].

Realistic advantages can be found compared to the statistical approach:

1. The error structure is quite simple and similar information usually available in most practical cases, not assuming *any a priori* statistical information about the error.
2. The computation of the parameter domain is conceptually simple and is practically feasible even if the number of data n is not large.
3. The algorithm is *deterministic*.

C. Constraint propagation on α-cut intervals

A 2-parameter problem will be used as an illustrative example, which will make it possible to draw pictures of the paving obtained. This example is a simplified version of a problem considered in [9][2]. Let X, Y two fuzzy sets and f a mapping from X to Y parametrized by $\vec{\theta}$, $\mu_X^\alpha(x)$ and $\mu_Y^\alpha(y)$ their respective characteristic functions for each $\alpha \in [0,1]$. In this simulated example, the prior intervals $[\check{x}_i]$ for $i = 1, \ldots, 10$ are obtained by adding the interval $[-2, 2]$ to the associated measurement x_i. The $[\check{y}_i]$ have been computed by adding a noise centered error interval with radius $\rho_i = 0,5w([x_i])$ or $\rho_i' = 0,5w([x_i])$ to the ith components of the data vector $\vec{y} = (1,2,3,4,5,6,7,8,9,10)^T$ and $\vec{x} = (4,3,3,2,2,2,1,1,1,1)^T$ for $i = 1, \ldots, n$. $[\check{x}_i]$,$[\check{y}_i]$ represent the $zero-$cut support of X and Y respectively with, says, triangle shape membership function (see Fig.2.a). The numerical values of the corresponding interval data are given in Table III-C. $[\check{x}_i^\alpha]$,$[\check{y}_i^\alpha]$ are crisp sets associated to $\mu_X^\alpha(x)$ and $\mu_Y^\alpha(y)$ respectively. The uncertainty associated with each pair ot input-output data is materialized by a set of card(α) *embeded boxes*, which size decreases as α increases. The set $\check{\Theta}$ to be characterized consists of all parameter vectors $\vec{\theta} = (\theta_1, \theta_2)^T$ such that the graph of the function

$$y = f(x) = \theta_1 e^{-\theta_2 x} \tag{12}$$

crosses all the ten boxes of Figure 7, where x, y, θ_1 and θ_2 are respectively the state variable[3], the output variable and the parameters of the model.

From Eq. (8) and (1), it comes that $\check{\Theta}$ is a fuzzy set defined by a characteristic function $\mu_{\check{\Theta}}^\alpha$ such that

$$\mu_{\check{\Theta}}^\alpha(\vec{\theta}) = \begin{cases} \alpha & \forall \vec{\theta} \in \check{\Theta}_\alpha \\ 0 & \text{otherwise} \end{cases}$$

where $\check{\Theta}_\alpha$ is the crisp set that consists of $\{\vec{\theta} \mid \mu_\Theta(\vec{\theta}) \geq \alpha\}$.

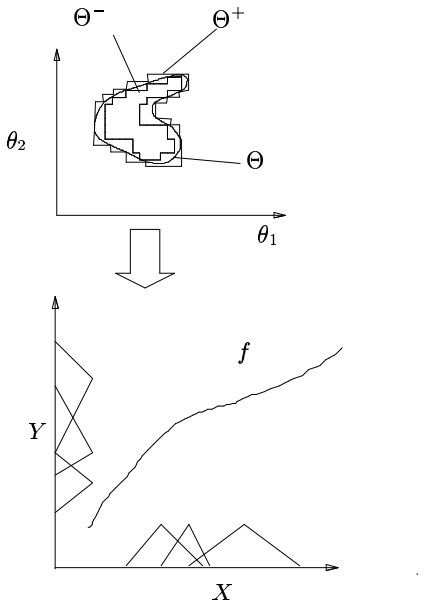


Fig. 6. Paving generated to bracket the posterior feasible set Θ for the parameters between inner and outer approximations.

[2]The extension of the method to multiple-output problems is straightforward.

[3]x and y can be any imprecise variables.

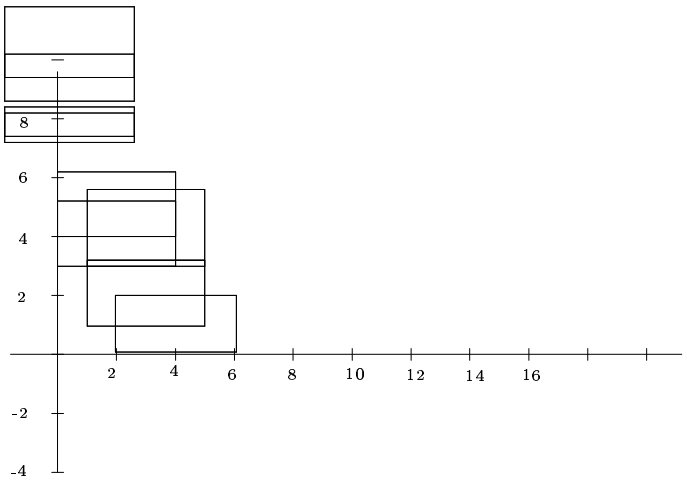


Fig. 7. Unprecise experimental data.

x_i	$[\check{x}_i]$	y_i	$[\check{y}_i]$
4	[2;6]	1	[0.13;2.1]
3	[1;5]	2	[1.04;3.51]
3	[1;5]	3	[2.65;4.87]
2	[0;4]	4	[3.12;5.23]
2	[0;4]	5	[4.06;6.16]
2	[0;4]	6	[5.7;7.5]
1	[-1;3]	7	[6.95;8.1]
1	[-1;3]	8	[7.57;9.38]
1	[-1;3]	9	[8.6;10.5]
1	[-1;3]	10	[9.1;11.02]

TABLE III

PRIOR FEASIBLE INTERVALS FOR THE DATA OF THE EXAMPLE.

The prior boxes for the parameters θ_1 and θ_2 may be arbitrarily large, by example

$$[\theta_1] = [-1000; 1000] \text{ and } [\theta_2] = [-1000; 1000],$$

i.e. no prior information is available on the parameters θ_1 and θ_2. According to Eq. 12, the equations to solve are, for each α-cut embeded box, the following:

$$\begin{cases} y_1^\alpha &= \theta_1^\alpha \exp(-\theta_2^\alpha x_1^\alpha) \\ &\vdots \\ y_{10}^\alpha &= \theta_1^\alpha \exp(-\theta_2^\alpha x_{10}^\alpha) \end{cases} \tag{13}$$

The algorithm given in Table II can be used to characterize $\check{\Theta}_\alpha$, provided that an inclusion test is available for

$$\exists \begin{pmatrix} x_1^\alpha \\ \vdots \\ x_{10}^\alpha \end{pmatrix} \in \begin{pmatrix} [\check{x}_1^\alpha] \\ \vdots \\ [\check{x}_{10}^\alpha] \end{pmatrix} \mid \begin{pmatrix} f(\vec{\theta}^\alpha, x_1^\alpha) \\ \vdots \\ f(\vec{\theta}^\alpha, x_{10}^\alpha) \end{pmatrix} \in Y^\alpha \text{ and } \begin{pmatrix} \theta_1^\alpha \\ \theta_2^\alpha \end{pmatrix} \in \begin{pmatrix} [\check{\theta}_1^\alpha] \\ [\check{\theta}_2^\alpha] \end{pmatrix} \tag{14}$$

i.e., as

$$(\vec{\theta}^\alpha \in \Theta_\alpha) \wedge \phi_1(\vec{\theta}^\alpha) \wedge \ldots \wedge \phi_{10}(\vec{\theta}^\alpha) \tag{15}$$

where $\phi_i(\vec{\theta}^\alpha)$ is the test

$$\phi_i(\vec{\theta}^\alpha) = (\exists x_i^\alpha \in [\check{x}_i] \mid f(x_i^\alpha, \vec{\theta}^\alpha) \in [\check{y}_i^\alpha]). \tag{16}$$

The problem of interest is to find crisp sets (if possible the smallest) enclosing variables θ_1, θ_2 that are consistent with the equations (13) and the domains in Table III-C. Jaulin and Walter in [9] reference their measures the same way. Hence, we have to define recursively the parameters region $(\theta_1^\alpha, \theta_2^\alpha)$ at the level α that generates output in the output interval $[y_i^\alpha]$ from the input $[x_i^\alpha]$. Arithmetic operations on fuzzy numbers can be derived in terms of arithmetic operations on their $\alpha-$cuts, which are arithmetic operations on *closed intervals* of real numbers. A value for any variable $\vec{\theta}$ is consistent with the problem if it is possible to instantiate $\vec{\theta}$ in their domains such that the relations (13) are satisfied. If all inconsistent values for the variables have been removed, the contraction will be said *optimal*. The algorithm which performs the contractions of the domains Θ_α alternates two types of interval propagation for each α-cut:

- a *forward propagation* step: it consists to run f with the interval inputs $[x_i^\alpha]$ and $[\theta_1^\alpha], [\theta_2^\alpha]$ using interval analysis. The new interval obtained for the y_i^α's will then be used to refine the $[y_i^\alpha]$'s.
- a *backward propagation* step: from the output intervals $[y_1^\alpha], \ldots, [y_{10}^\alpha]$, it sweeps f *backwardly* to refine $[\theta_1^\alpha], [\theta_2^\alpha]$. After completion of the backward propagation, the $[p_i^\alpha]$ have generally been contracted.

This decomposition is made in order to get only elementary operations, *i.e.* only one operator or function should be involved at each statement. Intermediate variables may be added. For our example, one gets

$$\begin{cases} z_2 &= \exp(\theta_1 x_i), \\ z_2 &= \theta_2 z_1, \\ y_i &= z_2. \end{cases}$$

The algorithm can be summarized as follow for each α:

Algorithm	
inputs :	$[x_i^\alpha], i = 1, \ldots, 10,\ [\theta_1^\alpha], [\theta_2^\alpha];$
init :	$[z_1] := [z_2] :=] -\infty; \infty[$
	repeat
1	$[z_1] := [z_1] \cap \exp([\theta_1] * [x_i^\alpha]);$
2	$[z_2] := [z_2] \cap [\theta_2^\alpha] * [z_1];$
3	$[\theta_1^\alpha] := [\theta_1^\alpha] \cap [y_i^\alpha]/[z_2];$
4	$[\theta_1^\alpha] := [\theta_1^\alpha] \cap \log([z_2])/[x_i^\alpha];$
5	$[z_1] := [z_1] \cap [z_2]/[\theta_1^\alpha];$
6	$[\theta_2^\alpha] := [\theta_2^\alpha] \cap \log([y_i^\alpha])/[z_1];$
	while the contraction is significant
output :	$[\theta_1^\alpha], [\theta_2^\alpha];$

Statements 1-2 intervalize the forward step, statements 3-6 intervalize the backward step. On the one hand, the algorithm attempts to partition X_α into subboxes $[\check{x}_i^\alpha]$ such that

$$f([\check{x}_i], [\vec{\theta}]) \cap Y_\alpha = \varnothing, \tag{17}$$

for all subboxes. All subboxes of X_α still to be studied are stored in the queue $\mathcal{Q}$. On the other hand, the algorithm tries to find one x such that

$$f([\check{x}],[\vec{\theta}]) \subset Y. \tag{18}$$

When such a x is found, this means that $\vec{\theta} \subset \mathbb{S}$. In practice, a test at step 6 is introduced to avoid splitting $[\check{x}_i]$ *ad ifinitum* and to introduce some relation with the splitting policy followed for Θ_α.

C.0.a Numerical example. Figure (III-C.0.a.a) represents the desired input/output relation given by Eq. (12). Figure (III-C.0.a.b) gives a fuzzy representation of the domain consistent with the set of equations (13). The plot is in the (θ_1, θ_2) space. The algorithm generates the subpavings represented on Fgure III-C.0.a.b in 2.4 seconds on a pentium III. The dark grey boxes have been proved to be included in $\mathbb{S}$ and the light grey boxes have been proved to have an empty intersection with $\mathbb{S}$. No conclusion has been reached for the black boxes. In this context, θ_1 and θ_2 determine *fuzzy numbers* which marginals are the membership functions of the parameters. The important point with such fuzzy identification method is that *no parameter solution is forgotten* during the iterative resolution.

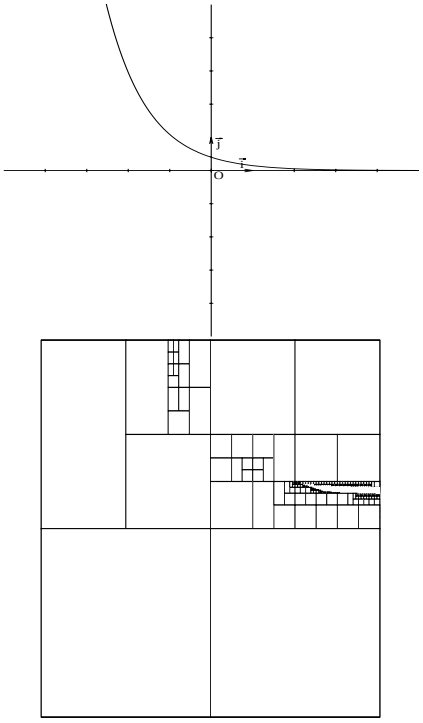


Fig. 8. a. Graph of the function f given by 12. b. Paving generated by the algorithm to bracket the posterior feasible set $\check{\Theta}$ for the parameters between inner and outer approximations. The outer frame corresponds to the box $[-0.614, 1.78]^{\times 2}$.

A 3-dimensional plot would show the fuzzy domain as a barn like structure rising up from its base. The approach considered so far relied on the hypothesis that the prior feasible set Θ did contain the actual values for the variables, which is unfortunately not always realistic.

IV. Conclusion

It is now possible to obtain guaranteed estimates of fuzzy parameters even when these parameters are not identifiable. Nonlinear constraints are easily handled. This approach could be extended to the case where X and Y consists of infinitely many variables, whatever could be the type of membership function to be used. Least square estimation suffers from the fact that the cost function to be minimized is a sum of terms involving the same parameters, so mutioccurence of these parameters is unavoidable and tends to make inclusion functions for the cost function very pessimistic, which complicates the elimination of interesting parts of the search domain.

The algorithm presented here has two major drawbacks. First, its complexity is exponential in the number of parameters, which restricts its use to low-dimensional problems. For the sake of simplicity, the algorithm presented here is far from optimal from the viewpoint of computational time, and significant improvements can be expected in the near future. Second, efficient functions are needed, which are available only when an explicit solution for the equations defining the model can be found.

The notion of subpaving introduced makes it possible to obtain, store and manipulate fuzzy approximations. Subpavings will form a usefull class of objects on which computations will be performed.

References

[1] Abe, Shigeo and Lan, M-S. (1995). Fuzzy rules extraction directly from numerical data for function approximation, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 25, No 1, 119-129.

[2] Bauman, E., Dorofeyuk, A. and Filev, D. (1990). Fuzzy identification of nonlinear dynamical systems, In *Proc. Int. Conf. on Fuzzy Logic and Neural Nets*, 895-898.

[3] Belforte, G., Bona, B. and Cerone, V. (1990). Parameter estimation algorithms for a set -memebership description of uncertainty, *Automatica*, Vol. 26, No. 5, pp. 887-898.

[4] Ishihashi, H. (1991). Iterative fuzzy modeling and a hierarchical network, Proc. 4th IFSA Congr., Brussels, Vol. Eng., pp. 49-52.

[5] Ishibuchi, H. and Nii, M. (2001), Numerical analysis of the learning of fuzzified neural networks from fuzzy if-then rules, *Fuzzy sets and Systems*, **120**, 281-307.

[6] Dickerson, J.A. and Kosko, B. (1996). Fuzzy function approximation with ellipsoidal rules, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 26, No 4, 542-560.

[7] Krishnamraju, P.V., Buckley, J.J., Reilly, K.D. and Hayashi, Y. (1994). Genetic learning algorithm for fuzzy neural nets. Proceedings of 1994 IEEE International Conference on Fuzzy Systems, pp. 1969-1974.

[8] Jaulin, L. (1994). Solution globale et garantie de problmes ensemblistes, application à l'estimation non-linéaire et à la commande robuste, thèse de l'université Paris-Sud.

[9] Jaulin, L. and Walter, E. (1999). Guaranteed bounded-error parameter estimation for nonlinear models with uncertain experimental factors. *Automation*, **35**, 849-856.

[10] Jaulin, L., Kieffer, M., Didrit, O. and Walter, E. (2001). *Applied interval analysis*, Springer, Paris.

[11] Mandani, E.H. (1974). Application of fuzzy algorihtms for control of simple dynamic plant, *Fuzzy sets and Systems*, **28**, 1858-1888.

[12] Nomura, H. Hayashi, I. and Wakami, N. (1991). A self-tuning method of fuzzy control by descent method, a hierarchical network, Proc. 4th IFSA Congr., Brussels, Vol. Eng., pp. 155-158.

[13] Norton, J.P. (1994). Special issue on bounded-error estimation: Issue 1. In *International Symposium Control and Signal Processing*, **8**(1), pp. 1-118.

[14] Sugeno, M. (1988). Structure identification of fuzzy models, *Fuzzy Sets Syst.*, Vol. 28, pp. 15-33.

[15] Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its application to modelling and control, In *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 15, n^{0}1, 116-132.

[16] Tay, T.T. and Tan, S.W. (1997). Fuzzy system as parameter estimator of nonlinear dynamic functions, In *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 27, n^{0}2, 313-322.

[17] Walter, E. (Eds) (1990). Special issue on parameter identification with error bounds. *Mathematics and Computers in Simulation* , **32**(5 et 6), pp. 447-607.

[18] Wang, L.X. and Mendel, J.M. (1992). Back-propagation fuzzy system as nonlinear dynamic system identifiers, Proc. IEEE Int. Conf. on Fuzzy Systems, San Diego, pp. 1409-1416.

[19] Yager, R.R. and Filev, D.P. (1993). Unified structure and parameter identification, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 23, No 4, 1198-1205.